

NGINX Conf 2018

The official event for all things NGINX



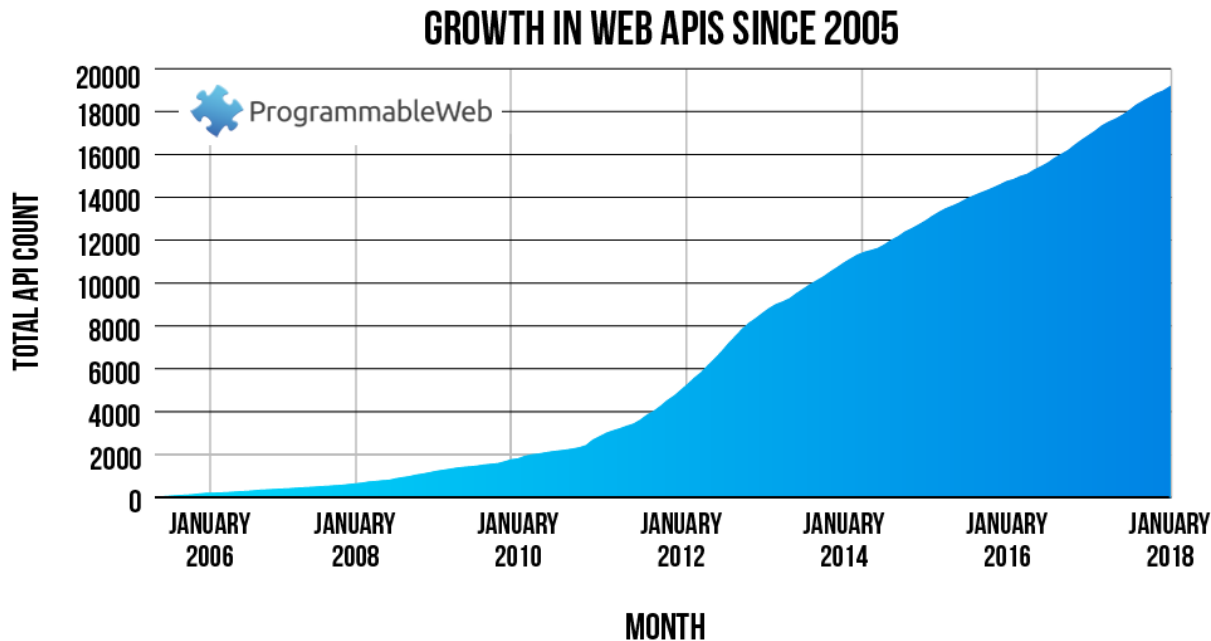
10 Tips for Deploying NGINX as an API Gateway

Liam Crilly
NGINX, Inc.





everything.io





Definitions





EXIT

NESN



Exception handling

AVILION C



TODAY'S PITCHERS:
COL: Hanrahan
DUR: Sonnenstine

COOPER STADIUM

TICKETS ARE NOT TO
BE SOLD OR DISTRIBUTED
ON THE GROUNDS
OF COOPER STADIUM

TICKET PRICES

BOX
CLUB
RESERVED

GENERAL ADMISSION

ADULT
YOUTH (12 & under)
SENIOR (65 & over)

THE COLLEGE CLIPPERS WISH TO TAKE
THIS OPPORTUNITY TO ADVISE ALL FANS
OF THE POTENTIAL RISK OF INJURY
FROM BATTERED OR THROWN BALLS,
BATS, AND OTHER OBJECTS OF THE
GAME. PLEASE BE ALERT AT ALL TIMES
AND KEEP YOUR EYES ON THE FIELD OF
PLAY. THANK YOU FOR COMING OUT TO
SUPPORT THE CLIPPERS.
ENJOY THE GAME!



BOX SEATS
CLUB
RESERVED
GENERAL ADMISSION

TICKET PRICES

BOX
CLUB
RESERVED

GENERAL ADMISSION

ADULT
YOUTH (12 & under)
SENIOR (65 & over)

THE COLLEGE CLIPPERS WISH TO TAKE
THIS OPPORTUNITY TO ADVISE ALL FANS
OF THE POTENTIAL RISK OF INJURY
FROM BATTERED OR THROWN BALLS,
BATS, AND OTHER OBJECTS OF THE
GAME. PLEASE BE ALERT AT ALL TIMES
AND KEEP YOUR EYES ON THE FIELD OF
PLAY. THANK YOU FOR COMING OUT TO
SUPPORT THE CLIPPERS.
ENJOY THE GAME!



BOX SEATS
CLUB
RESERVED
GENERAL ADMISSION

TICKETS

TICKET PRICES

BOX
CLUB
RESERVED

GENERAL ADMISSION

ADULT
YOUTH (12 & under)
SENIOR (65 & over)

THE COLLEGE CLIPPERS WISH TO TAKE
THIS OPPORTUNITY TO ADVISE ALL FANS
OF THE POTENTIAL RISK OF INJURY
FROM BATTERED OR THROWN BALLS,
BATS, AND OTHER OBJECTS OF THE
GAME. PLEASE BE ALERT AT ALL TIMES
AND KEEP YOUR EYES ON THE FIELD OF
PLAY. THANK YOU FOR COMING OUT TO
SUPPORT THE CLIPPERS.
ENJOY THE GAME!

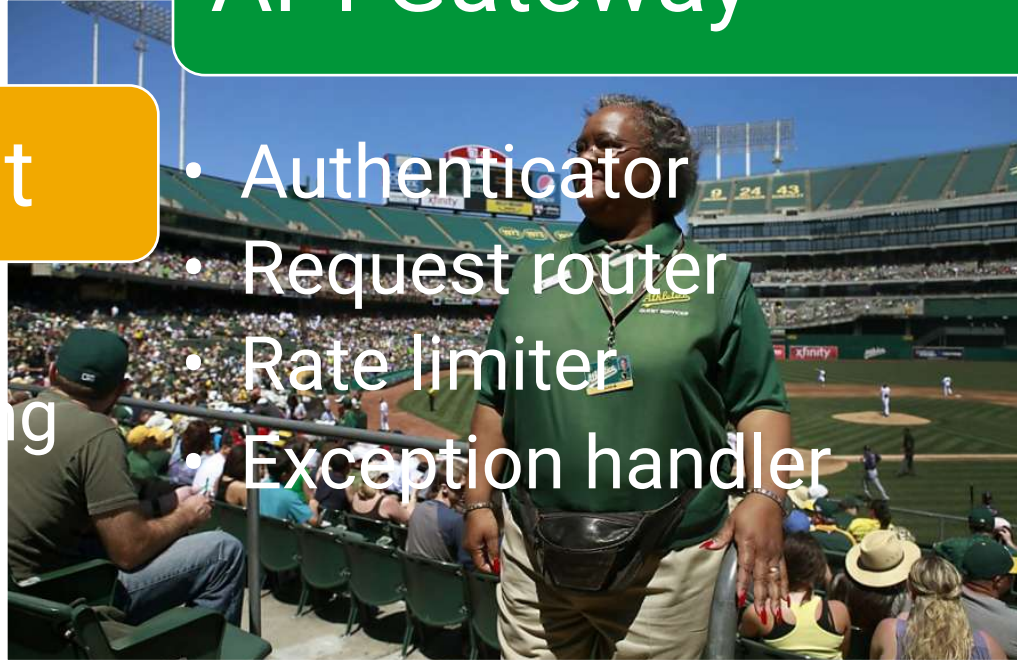
TICKETS & WILL CALL (PHONE & INTERNET)



API Gateway

API Management

- Policy management
- Analytics & monitoring
- Developer documentation



- Authenticator
- Request router
- Rate limiter
- Exception handler

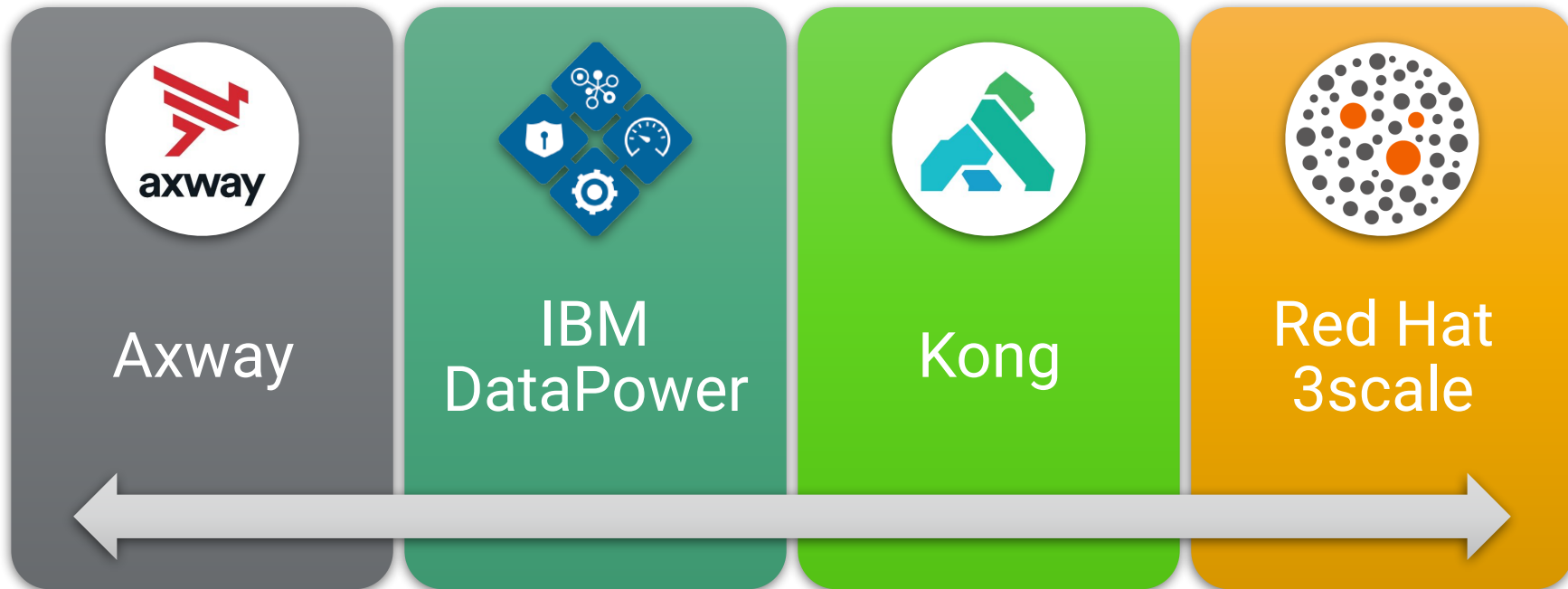
40%

NGINX Plus deployed as an API gateway



Source: NGINX brand survey 2017, 2018

NGINX as an API Gateway





Ergast Developer API

jcnnewell/ergast-f1-api: A PHP-based read-only API for the Ergast Formula One MySQL database

GitHub, Inc. [US] | https://github.com/jcnnewell/ergast-f1-api

<> Code

🔔 Issues 3

🔗 Pull requests 1

📁 Projects 0

📖 Wiki

📊 Insights

A PHP-based read-only API for the Ergast Formula One MySQL database

🔄 9 commits

🌿 1 branch

📦 0 releases

👤 2 contributors

🔧 Apache-2.0

Branch: master ▾

New pull request

Create new file

Upload files

Find file

Clone or download ▾

 jcnnewell Merge pull request #2 from psychemedia/master ...

Latest commit 62ba8d8 on 5 Sep 2017

📁 ergastdb	Support for docker-compose execution	a year ago
📁 lamp	Support for docker-compose execution	a year ago
📁 webroot	Support for docker-compose execution	a year ago
📄 LICENSE	Initial commit	2 years ago
📄 README.md	Remember to wait a mo...	a year ago
📄 docker-compose.yaml	Support for docker-compose execution	a year ago

📖 README.md

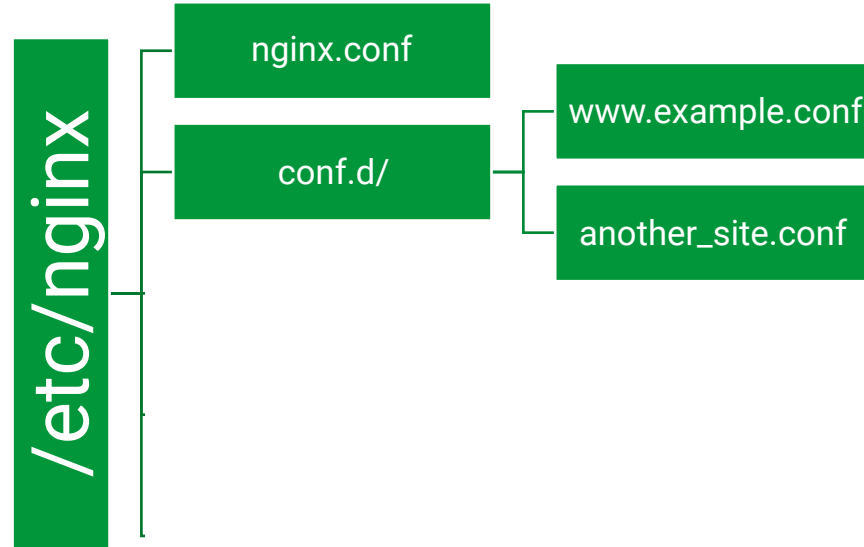
ergast Local API Server

Based on the [ergast-f1-api](#), a PHP-based API using the [Ergast Formula One MySQL database](#) developed by Chris Newell.

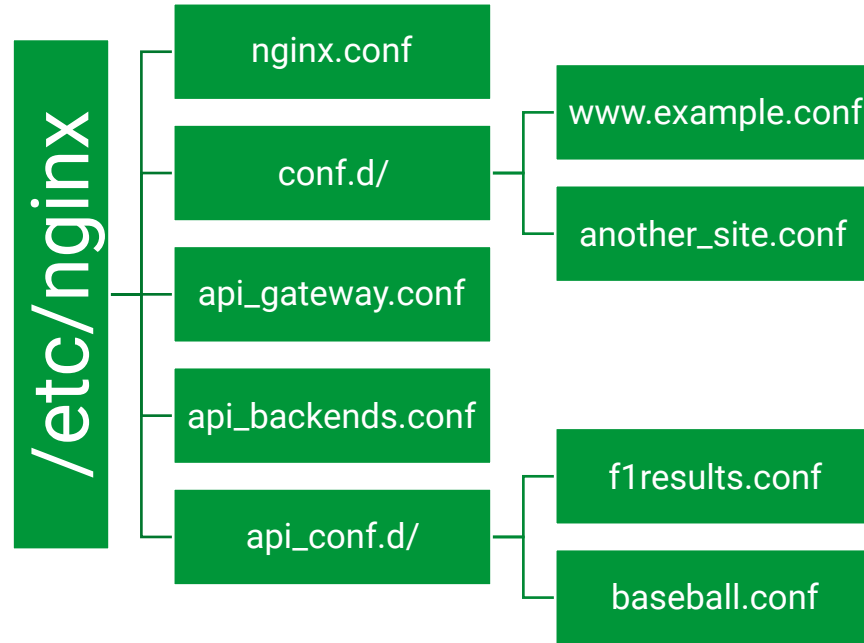


1. Organize config

1. Organize configuration



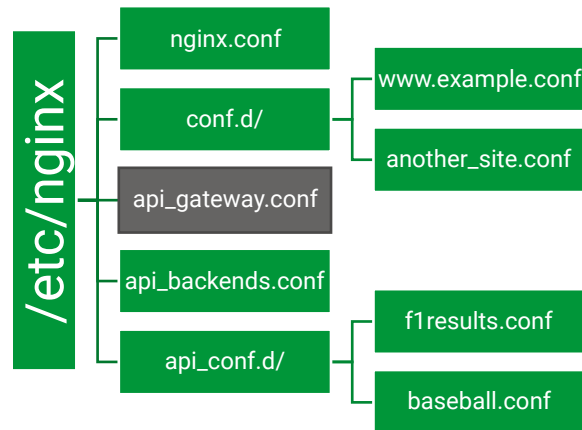
1. Organize configuration



2. Single client-facing hostname

api_gateway.conf

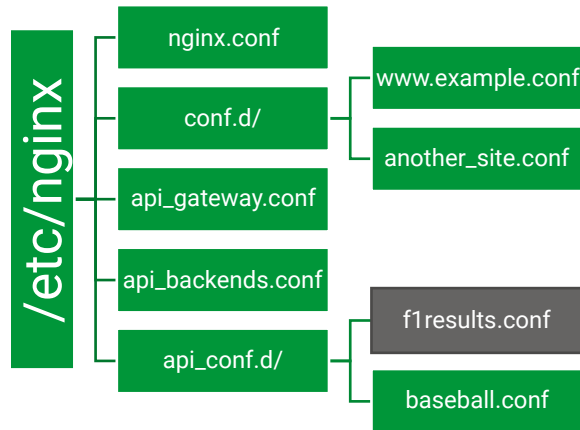
```
1 log_format api_main '$remote_addr - $remote_user [$time_local] "$request" '
2   '$status $body_bytes_sent "$http_referer" '
3   '"$http_user_agent" "$http_x_forwarded_for" "$api_name"';
4
5 include api_backends.conf;
6
7 server {
8     set $api_name -; # Start with an undefined API name
9     access_log log/api_access.log api_main;
10
11     listen 443 ssl;
12     server_name api.example.com;
13
14     ssl_certificate      /etc/nginx/ssl/api.example.com.crt;
15     ssl_certificate_key  /etc/nginx/ssl/api.example.com.key;
16
17     include api_conf.d/*.conf;
18 }
```



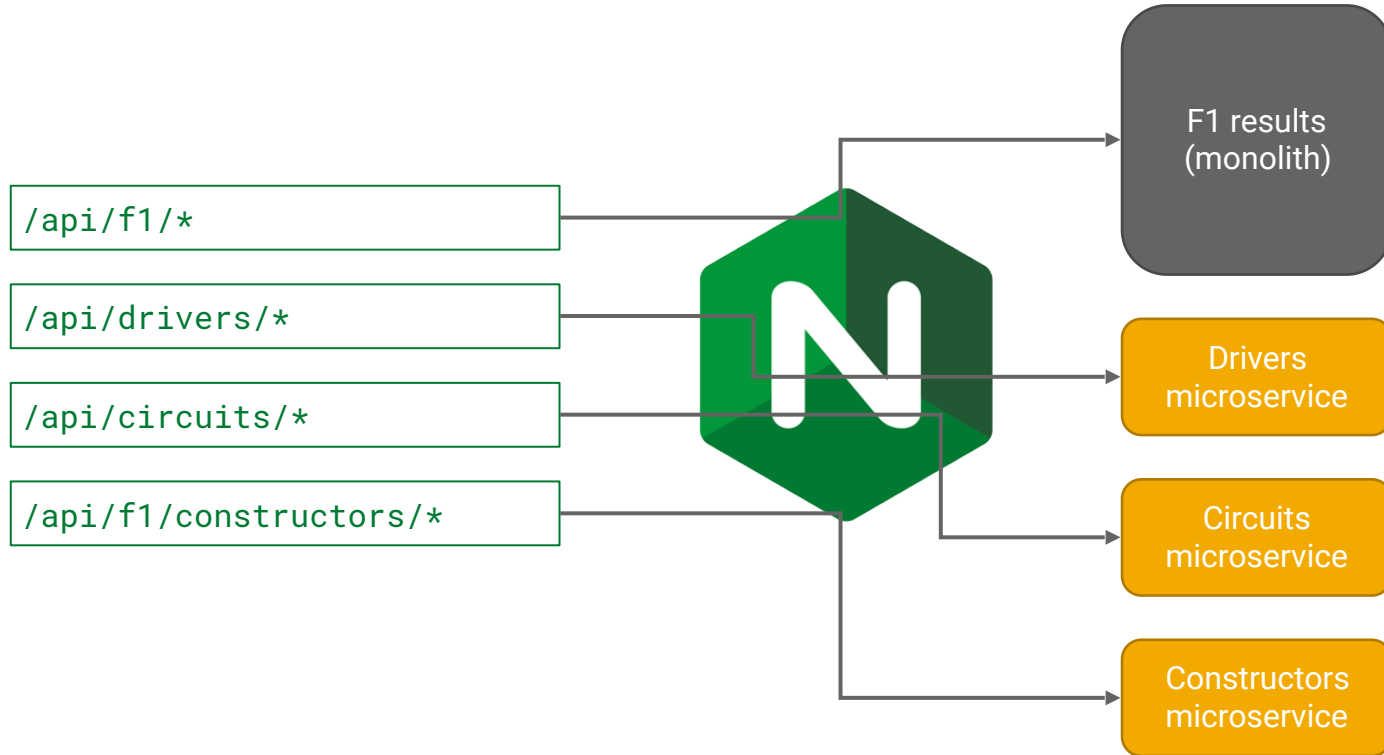
3. Separate definition from policy

f1results.conf

```
1 # API definition
2 #
3 location = /api/f1 { rewrite ^ /_ergast last; }
4 location ~ /api/f1/[12][0-9]+ { rewrite ^ /_ergast last; }
5 location /api/f1/current { rewrite ^ /_ergast last; }
6 location /api/f1/seasons { rewrite ^ /_ergast last; }
7 location /api/f1/status { rewrite ^ /_ergast last; }
8 location /api/f1/drivers { rewrite ^ /_ergast last; }
9 location /api/f1/circuits { rewrite ^ /_ergast last; }
10 location /api/f1/constructors { rewrite ^ /_ergast last; }
11
12 # Policy section
13 #
14 location = /_ergast {
15     internal;
16     set $api_name "F1 results";
17
18     # Policy configuration here (authentication, rate limiting, logging, ...)
19
20     proxy_pass http://ergast_api$request_uri.json;
21 }
```



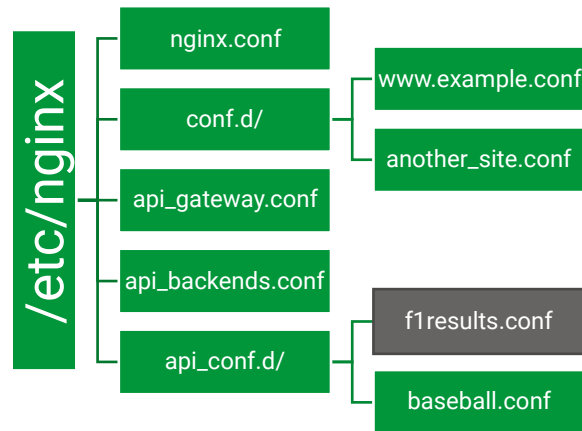
4. Microservices routing



4. Microservices routing

f1results.conf

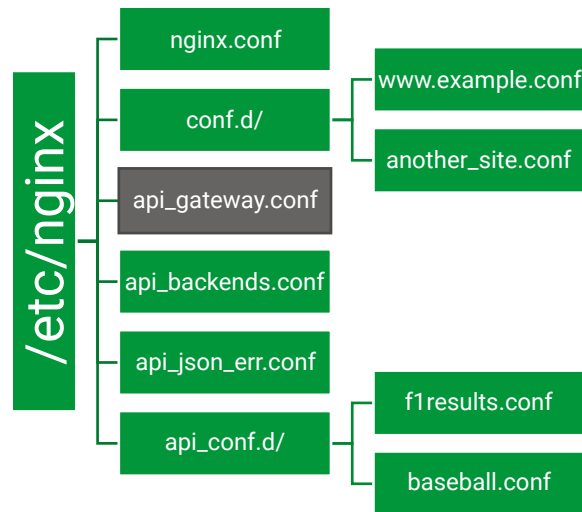
```
1 # API definition
2 #
3 location /api/f1          { rewrite ^ /_ergast_mono last; }
4 location /api/f1/drivers  { rewrite ^ /_ergast_drivers last; }
5 location /api/f1/circuits { rewrite ^ /_ergast_circuits last; }
6 location /api/f1/constructors { rewrite ^ /_ergast_constructors last; }
7
8 # Routing
9 #
10 location = /_ergast_mono {
11     proxy_pass http://f1_mono_api$request_uri.json;
12 }
13 location = /_ergast_drivers {
14     proxy_pass http://f1_drivers_service$request_uri.json;
15 }
16 location = /_ergast_circuits {
17     proxy_pass http://f1_circuits_service$request_uri.json;
18 }
19 location = /_ergast_constructors {
20     proxy_pass http://f1_constructors_service$request_uri.json;
21 }
```



5. Send appropriate errors

api_gateway.conf

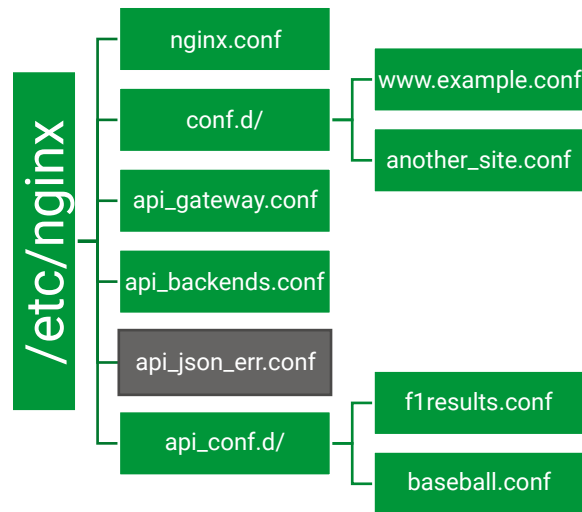
```
7 server {
8     set $api_name -; # Start with an undefined API name
9     access_log log/api_access.log api_main;
10
11     listen 443 ssl;
12     server_name api.example.com;
13
14     ssl_certificate      /etc/nginx/ssl/api.example.com.crt;
15     ssl_certificate_key  /etc/nginx/ssl/api.example.com.key;
16
17     include api_conf.d/*.conf;
18
19     # Error responses
20     location / {
21         return 404;                # Catch-all for unrecognized URIs
22     }
23     error_page 404 = @400;         # Invalid paths are treated as bad requests
24     proxy_intercept_errors on;    # Do not send endpoint errors to the client
25     include api_json_err.conf;   # API client friendly JSON error responses
26     default_type application/json; # If no content-type then assume JSON
27 }
```



5. Send appropriate JSON errors

api_json_err.conf

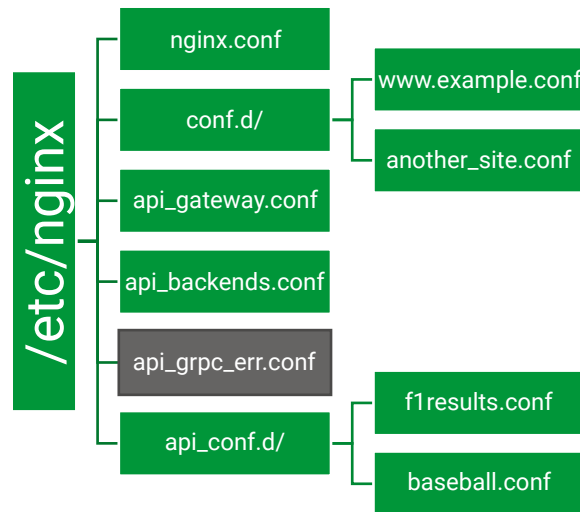
```
1 error_page 400 = @400;
2 location @400 { return 400 '{"status":400,"message":"Bad request"}\n'; }
3
4 error_page 401 = @401;
5 location @401 { return 401 '{"status":401,"message":"Unauthorized"}\n'; }
6
7 error_page 403 = @403;
8 location @403 { return 403 '{"status":403,"message":"Forbidden"}\n'; }
9
10 error_page 404 = @404;
11 location @404 { return 404 '{"status":404,"message":"Resource not found"}\n'; }
12
13 error_page 405 = @405;
14 location @405 { return 405 '{"status":405,"message":"Method not allowed"}\n'; }
15
16 error_page 408 = @408;
17 location @408 { return 408 '{"status":408,"message":"Request timeout"}\n'; }
18
19 error_page 413 = @413;
20 location @413 { return 413 '{"status":413,"message":"Payload too large"}\n'; }
21
22 error_page 414 = @414;
23 location @414 { return 414 '{"status":414,"message":"URI too large"}\n'; }
```



5. Send appropriate gRPC errors

api_grpc_err.conf

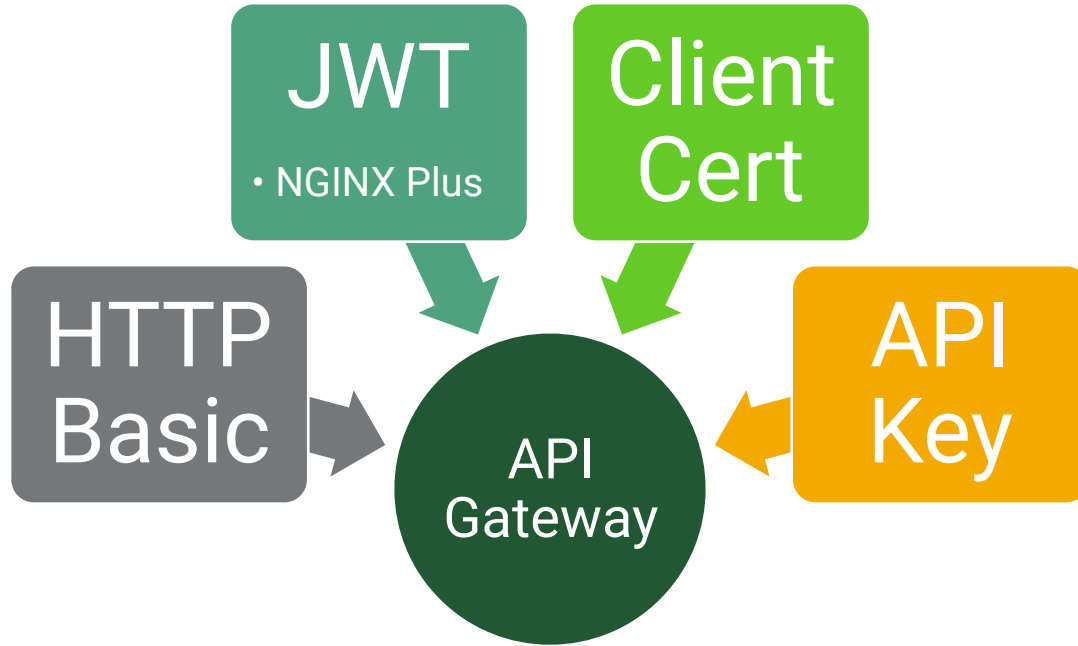
```
1 # Standard HTTP-to-gRPC status code mappings
2 # Ref: https://github.com/grpc/grpc/blob/master/doc/http-grpc-status-mapping.md
3 #
4 error_page 400 = @grpc_internal;
5 error_page 401 = @grpc_unauthenticated;
6 error_page 403 = @grpc_permission_denied;
7 error_page 404 = @grpc_unimplemented;
8 error_page 429 = @grpc_unavailable;
9 error_page 502 = @grpc_unavailable;
10 error_page 503 = @grpc_unavailable;
11 error_page 504 = @grpc_unavailable;
12
13 # gRPC error responses
14 # Ref: https://github.com/grpc/grpc-go/blob/master/codes/codes.go
15 #
16 location @grpc_internal {
17     add_header grpc-status 13;
18     add_header grpc-message 'internal error';
19     return 204;
20 }
21
22 location @grpc_unauthenticated {
23     add_header grpc-status 16;
24     add_header grpc-message unauthenticated;
25     return 204;
26 }
```





6. Limit authentication options

6. Limit authentication options



7. Protect API keys

api_gateway.conf

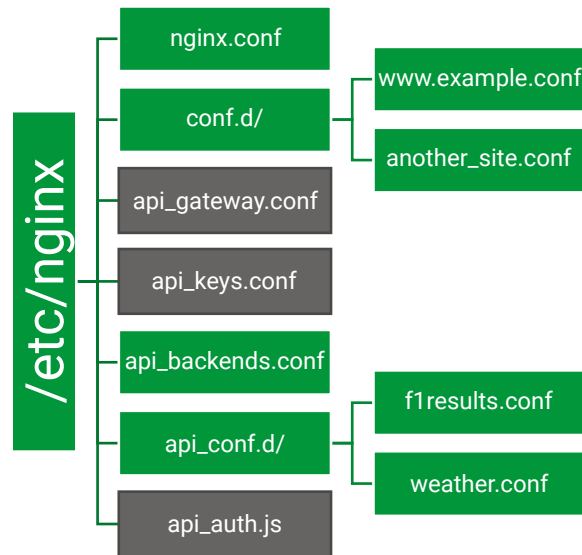
```
29 include api_keys.conf
30
31 js_include api_auth.js;
32 js_set $apikey_hash hashApiKey;
```

api_keys.conf

```
1 map $apikey_hash $api_client_name {
2     default "";
3
4     "vvV+x/U6bUC+tkCngKY5yDvCmsipgW8fxsXG3Nk8RyE=" "client_one"; #abcdef
5     "P7edLn93QUtR2PQd97EEKZEQkR1Kke0vU/rw1Je7gY0=" "client_two"; #beefed
6     "yorH9e2lUcrLrgIz91DHIEBDYxy8YenXoN1pLlncVKo=" "client_three"; #fedcab
7 }
```

api_auth.js

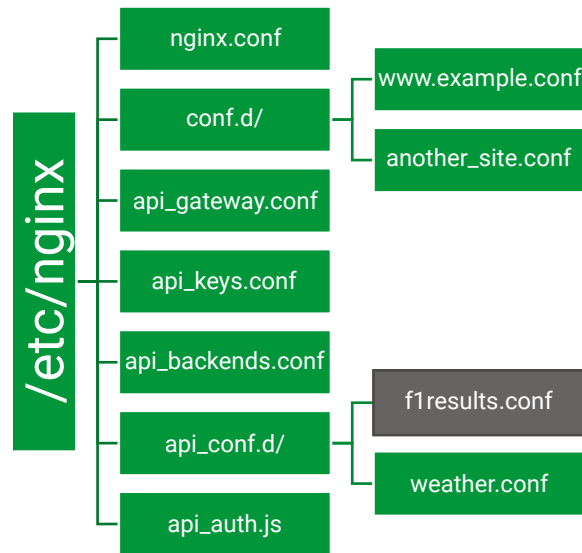
```
1 function hashApiKey(req) {
2     var c = require('crypto');
3     var h = c.createHash('sha256').update(req.variables.http_apikey);
4     return h.digest('base64');
5 }
```



7. Protect API keys

f1results.conf

```
1 # API definition
2 #
3 location /api/f1          { rewrite ^ /_ergast last; }
4 location /api/f1/drivers  { rewrite ^ /_ergast last; }
5 location /api/f1/circuits { rewrite ^ /_ergast last; }
6 location /api/f1/constructors { rewrite ^ /_ergast last; }
7
8 # Policy section
9 #
10 location = /_ergast {
11     internal;
12     set $api_name "F1 results";
13
14     # Policy configuration here (authentication, rate limiting, logging, ...)
15     if ($http_apikey = "") {
16         return 401; # Unauthorized (please authenticate)
17     }
18     if ($api_client_name = "") {
19         return 403; # Forbidden (invalid API key)
20     }
21
22     proxy_pass http://ergast_api$request_uri.json;
23 }
```



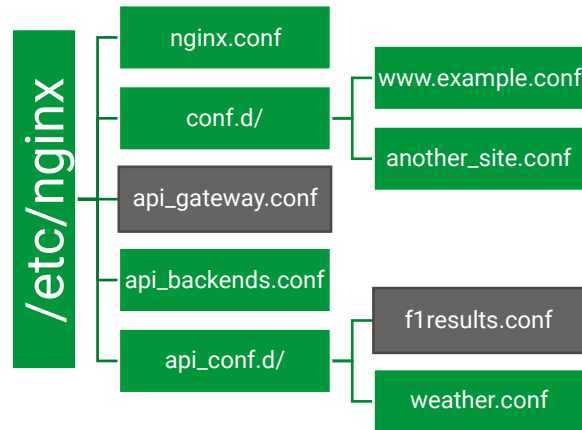
8. Know why you're rate limiting

api_gateway.conf

```
1 log_format api_main '$remote_addr - $remote_user [$time_local] "$request" '
2                       '$status $body_bytes_sent "$http_referer" '
3                       '"$http_user_agent" "$http_x_forwarded_for" "$api_name"';
4
5 include api_backends.conf;
6
7 limit_req_zone $binary_remote_addr zone=client_ip_10rps:1m rate=10r/s;
8 limit_req_zone $http_apikey zone=apikey_200rps:1m rate=200r/s;
```

f1results.conf

```
12 # Policy section
13 #
14 location = /_ergast {
15     internal;
16     set $api_name "F1 results";
17
18     limit_req zone=apikey_200rps;
19     limit_req zone=client_ip_10rps;
20     limit_req_status 429; # Too many requests
21
22     proxy_pass http://ergast_api$request_uri.json;
23 }
```





9. Inspect the request method

API Dimensions

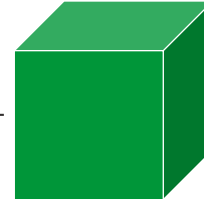

/api/f1/seasons

POST



/api/f1/seasons

POST



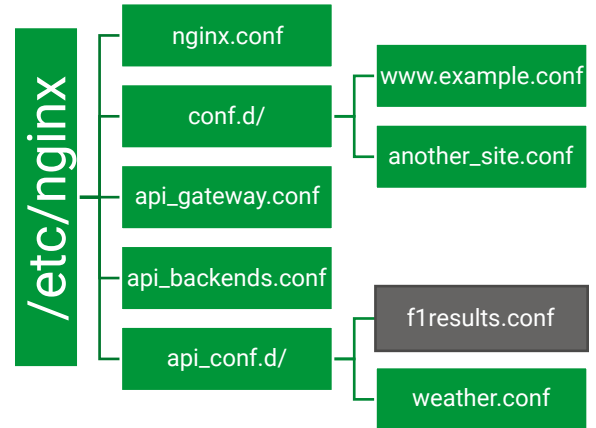
/api/f1/seasons

```
{  
  "season": 2019,  
  "url": "http://..."  
}
```

9. Inspect the request method

f1results.conf

```
1 # API definition
2 #
3 location /api/f1 {
4     limit_except GET { deny all; }
5     rewrite ^ /_ergast last;
6 }
7
8 location = /api/f1/seasons {
9     limit_except GET POST { deny all; }
10    if ( $request_method = POST ) {
11        rewrite ^ /_ergast_masterdb last;
12    }
13    rewrite ^ /_ergast last;
14 }
15
16 # Policy/routing section
17 #
18 location = /_ergast {
19     internal;
20     proxy_pass http://ergast_api$request_uri.json;
21 }
22
23 location = /_ergast_masterdb {
24     internal;
25     proxy_pass http://10.0.0.1$request_uri;
26 }
```



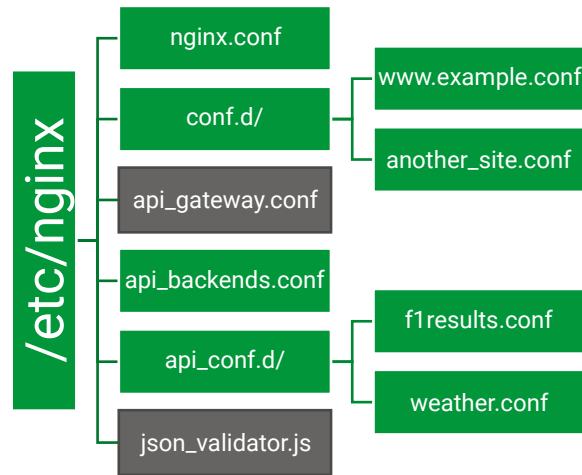
10. Inspect body if necessary

api_gateway.conf

```
29 js_include json_validator.js;
30 js_set $validated json_validator;
31
32 server {
33     listen 127.0.0.1:10415; # This is the error response of json_validator()
34     return 415; # Unsupported media type
35     include api_json_err.conf;
36 }
```

json_validator.js

```
1 function json_validator(req) {
2     try {
3         if (req.variables.request_body.length > 0) {
4             JSON.parse(req.variables.request_body);
5         }
6         return req.variables.upstream;
7     } catch (e) {
8         req.log('JSON.parse exception');
9         return '127.0.0.1:10415';
10    }
11 }
```



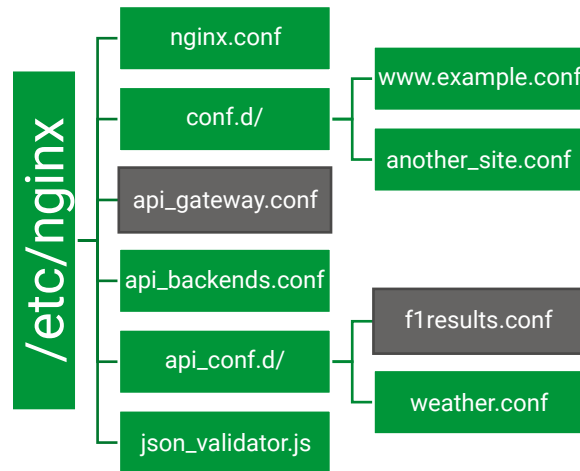
10. Inspect body if necessary

f1results.conf

```
23 location = /_ergast_masterdb {
24     internal;
25     set $api_name "Ergast MRD";
26     set $upstream ergast_api_master;
27
28     mirror /_null;
29     client_body_in_single_buffer on;
30     client_max_body_size 16k;
31
32     proxy_pass http://$validated;
33 }
```

api_gateway.conf

```
29 # Dummy location used to populate $request_body for JSON validation
30 location /_null {
31     internal;
32     return 204;
33     access_log off;
34 }
```

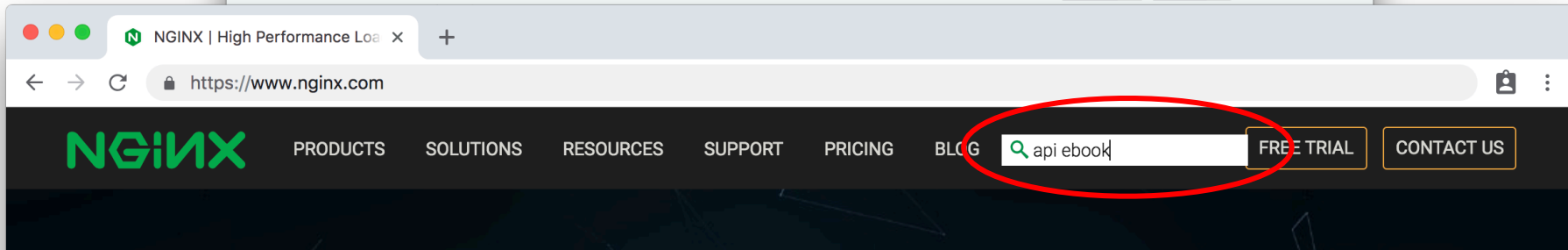
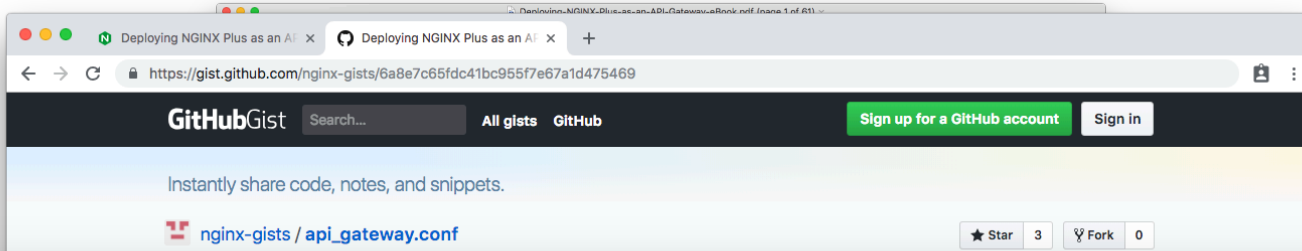


EXIT

NESN



AVILION C



```
6 include api_keys.conf;
7
8 limit_req_zone $binary_remote_addr zone=client_ip_10rs:1m rate=10r/s;
9 limit_req_zone $http_apikey zone=apikey_200rs:1m rate=200r/s;
10
11 server {
12     set $api_name ~; # Start with an undefined API name, each API will update this value
13     access_log /var/log/nginx/api_access.log api_main; # Each API may also log to a separate file
14
15     listen 443 ssl;
16     server_name api.example.com;
17
18     # TLS config
19     ssl_certificate /etc/ssl/certs/api.example.com.crt;
20     ssl_certificate_key /etc/ssl/private/api.example.com.key;
21     ssl_session_cache shared:SSL:10m;
22     ssl_session_timeout 5m;
23     ssl_ciphers HIGH:!aNULL:!MD5;
24     ssl_protocols TLSv1.1 TLSv1.2;
25
26     # API definitions, one per file
27     include api_conf.d/*.conf;
```

NGINX

fin

<https://www.nginx.com/resources/library/nginx-api-gateway-deployment/>

@liamcrilly
liam@nginx.com





Q&A

<https://www.nginx.com/resources/library/nginx-api-gateway-deployment/>

@liamcrilly
liam@nginx.com

